DOCUMENT RESUME

ED 479 431 IR 022 108

AUTHOR	Baudry, Andreas; Bungenstock, Michael; Mertsching, Barbel
TITLE	Architecture of an E-Learning System with Embedded Authoring Support.
SPONS AGENCY	Federal Ministry of Education, Science, Research, and Technology, Bonn (Germany).
PUB DATE	2002-10-00
NOTE	8p.; In: E-Learn 2002 World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education. Proceedings (7th, Montreal, Quebec, Canada, October 15-19, 2002); see IR 022 093.
CONTRACT	08NM084
AVAILABLE FROM	Association for the Advancement of Computing in Education (AACE), P.O. Box 3728, Norfolk, VA 23514. Tel: 757-623-7588; e-mail: info@aace.org; Web site: http://www.aace.org/DL/.
PUB TYPE	Reports - Research (143) -- Speeches/Meeting Papers (150)
EDRS PRICE	EDRS Price MF01 Plus Postage. PC Not Available from EDRS
DESCRIPTORS	Authoring Aids (Programming); *Computer Assisted Instruction; Computer Software Development; Computer System Design; *Distance Education; *Educational Resources; Higher Education; Mathematics Education; *Multimedia Materials; Undergraduate Study

ABSTRACT

This paper introduces an architecture for an e-learning system with an embedded authoring system. Based on the metaphor of a construction kit, this approach offers a general solution for specific content creation and publication. The learning resources are IMS "Content Packages" with a special structure to separate content and presentation. These packages are called "Template Content Objects" (TCO). They contain a template coded in XML and additional files for multimedia learning objects. TCOs are similar to the SCORM "Sharable Content Objects" (SCO) and can be simply translated to them. The authoring system consists of different authoring tools that handle TCOs. Every task in the process of learning object creation is done by experts which were modeled with the help of roles. The architecture is a result of the math-kit, a Web based e-learning system that provides instructors and students with multimedia support for central topics in undergraduate mathematics. (Contains 17 references.) (Author)

Reproductions supplied by EDRS are the best that can be made
from the original document.

Architecture of an E-Learning System with Embedded Authoring Support

PERMISSION TO REPRODUCE AND DISSEMINATE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY

G.H. Marks

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

2B

Andreas Baudry, Michael Bungenstock, Bärbel Mertsching
IMA Lab
Dept. of Informatics
University of Hamburg
Germany
baudry@informatik.uni-hamburg.de

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

This document has been reproduced as received from the person or organization originating it.

Minor changes have been made to improve reproduction quality.

Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

Abstract: In this paper we introduce an architecture for an e-learning system with an embedded authoring system. Based on the metaphor of a construction kit, this approach offers a general solution for specific content creation and publication. The learning resources are IMS "Content Packages" with a special structure to separate content and presentation. We call these packages "Template Content Objects" (TCO). They contain a template coded in XML and additional files for multimedia learning objects. TCOs are similar to the SCORM "Sharable Content Objects" (SCO) and can be simply translated to them. The authoring system consists of different authoring tools that handle TCOs. Every task in the process of learning object creation is done by experts which we modeled with the help of roles. The architecture is a result of the math-kit project, a web based e-learning system that provides instructors and students with multimedia support for central topics in undergraduate mathematics.

ED 479 431

Introduction

Our goal is the development of a multimedia construction kit, called „math-kit", that supports the learning and teaching of mathematical and technical content (Mathkit 2002, Unger 2002). Hence, the math-kit project provides a mechanism to find and combine learning objects to compose them to complete learning units and online courses. Students will be able to use math-kit to work through online courses to refresh their knowledge or to use the tools for further exploration of course material. In the same way the students use courses, instructors could apply parts of the courses to their lectures to make them more accesible to students.

Math-kit is a joint project, which is being developed at the German universities of Hamburg, Bayreuth, Hagen and Paderborn[1]. They combine their technical and didactic skills to produce mathematical and technical learning units for interdisciplinary use in higher education.

In a joint project, the creation of uniform learning content is very difficult to achieve. Often, different developers produce learning objects which are based on varying description languages such as HTML or PDF. The software market provides several tools for content creation, for instance, Microsoft Word, Adobe Acrobat, Adobe Framemaker or Macromedia Dreamweaver. Each of these programs provides special functionalities for specific problems but has weaknesses in other tasks. Also, defining rules for the style and layout of content is insufficient because the process of individual combination and the reuse of learning material in other contexts is not possible. For this purpose the architecture of math-kit uses the construction kit metaphor (Bungenstock 2002) that we have developed as a solution for this problem.

Briefly, this metaphor helps to plan a software system that supports both students and instructors in improving their work. To understand this metaphor, think of a LEGO construction kit with all contained bricks. The unique design of the bricks allows a multitude of stable models. In the math-kit context the learning objects are the bricks, which can be combined to individual courses. To define such a brick in a e-

[1]Support of the German Federal Ministry of Education and Research, project Math-Kit (08NM084) is gratefully acknowledged.

IR022108

BEST COPY AVAILABLE

learning system, we provide the TCO approach that uses the *Extensible Markup Language (XML)* (XML 2002) to separate content and layout of learning objects.

Template Content Object

Before we describe the architecture for the math-kit project in detail, a deeper understanding of our special template mechanism is required. Our *Template Content Object (TCO)* is inspired by the *Sharable Content Object (SCO)* of the SCORM standard (ADL 2002) which has a strong influence on our design. Briefly, a SCO represents a collection of different files, called assets, and is the atomic learning resource for a *Learning Management System (LMS)*. One SCO is intended as a small and independent unit to be reused in different learning contexts. All learning resources and additional metadata describing the SCO can be bundled by a special *Content Package* mechanism based on the standard of the IMS (IMS 2002). SCORM offers two content packaging application profiles that describe how IMS content packaging is used. Both *resource packages* and *content aggregation packages* use a *manifest* for the internal description and the same file encoding like zip compression. However, significant differences exist between them. A resource package contains only assets with no relations or order and is therefore the SCO. The content aggregation package consists of SCOs or other content aggregations and offers higher structuring. Hence, higher-level structures like lessons, sections and chapters, as well as complete courses, can be built in the same way. Figure 1 illustrates the levels of content packaging. The arrows show the inclusion relation.

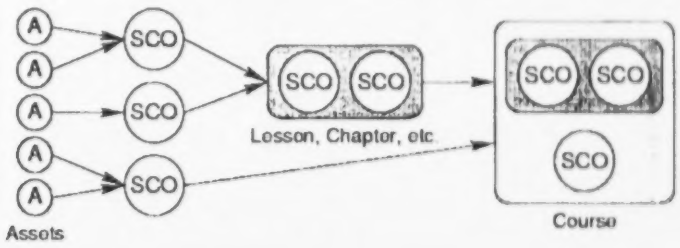

Figure 1: Recursive content packaging

In this simplified example, a course is a content aggregation package containing another content aggregation package and a resource package/SCO. The recursive definition of content aggregations allows any structure depth for learning content desired.

For assessment of learners, SCORM defines a simple protocol for comunication between SCO and LMS. At runtime, the SCO obtains the API adapter of the hosting LMS for interchanging data about the learner's behaviour.

The SCO approach meets the demands for content interchange and execution but lacks some important features for content creation. As already mentioned, the separation of content and view is one of the essential goals of math-kit. Therefore, the *Extensible Markup Langugage (XML)* and the *Extensible Stylesheet Language Translations (XSLT)* (XSLT 2002) are used. A TCO has the same structure as a SCO with the exception of one special asset: a template written in XML. In the case of math-kit, the template describes the texts, mathematical definitions and formulas in a well-defined fashion. Templates can be used for every kind of content but they depend on a special XML encoding. The further assets in the TCO are required only for the later translation of the template into a readable output format, like HTML or PDF. To make the conversion clear, figure 2 shows an example with a TCO containing two images and one animation. The TCO is translated into two different SCOs whereby the varying number of files in the SCOs is caused by technical differences. The PDF file format allows the integration of images but HTML contains only references to other resources.

BEST COPY AVAILABLE

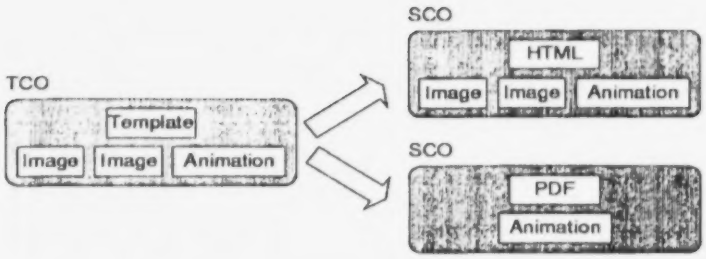

Figure 2: A TCO converted into two different SCOs

The translation process is controlled by different XSLT files which are used to convert XML into any hierarchical file format. Thus, the individual appearances of one TCO can be realized with only one translation technique. For instance, if you need to show the emblem of your company, school or university in the upper right-hand corner using particular colors, you only have to modify the XSLT files and not the template. This means that a created TCO never has to be changed for visual modification.

The Roles

We introduce roles to model the different views of our e-learning system. It is important to comprehend that one person can appear in several roles. The roles help us to model the static and dynamic behavior of our system with use cases and sequential diagrams (Bergner 1998). This allows us to have a strict and controllable development process.

Six different roles can be defined with math-kit: instructor, student, composer, learning object developer, publisher, and administrator. The *instructor* uses single blocks or complete courses to extend his own lectures or to create new ones. Important operations for this role are searching and archiving. The *student* role needs similar operations but for another purpose. Individuals in this role want to learn and can explore the entire database to use learning objects and courses of interest. These two roles don't depend directly on the construction kit metaphor because they work with math-kit through a LMS. Nevertheless, the construction kit is the foundation of the created content.

The composer uses math-kit to combine existing learning objects to make complete courses. Useful operations for this role include searching for learning objects, combining, previewing and perhaps editing. This role creates and edits the content package. In relation to the construction kit metaphor, the composer defines the construction plan. The *learning object developer* designs and implements new learning objects. This role consists of collecting and assembling the different assets like images, texts, and animations to SCOs. In view of the metaphor, the learning object developer creates several bricks. In context of math-kit, the learning content is coded in XML with a structure for mathematics. This XML code has to be translated into a readable format by the role *publisher*. Hence, the main operations of the publisher are the definition or choosing of particular styles and the compilation of the source document to a destination format. The last role, *administrator*, installs basic packages and software for the construction-kit. The main task of this role is to help all the other roles with technical problems. Maintenance, consulting and user administration are the important operations.

A little example should make the relations among the roles clear: Consider a professor who wants to extend an existing lecture in computer engineering with a multimedia course. First of all, the professor is in the role of the composer. Math-kit offers a wide range of content and most of the required themes ideally should be available. If some content is missing or has to be adapted the professor slips into the role of the learning object developer. For instance, the learning object developer is able to integrate the mathematical content of a colleague, which is not stored in math-kit. Figure 3 shows the role of the composer creating a course for computer engineering.

BEST COPY AVAILABLE

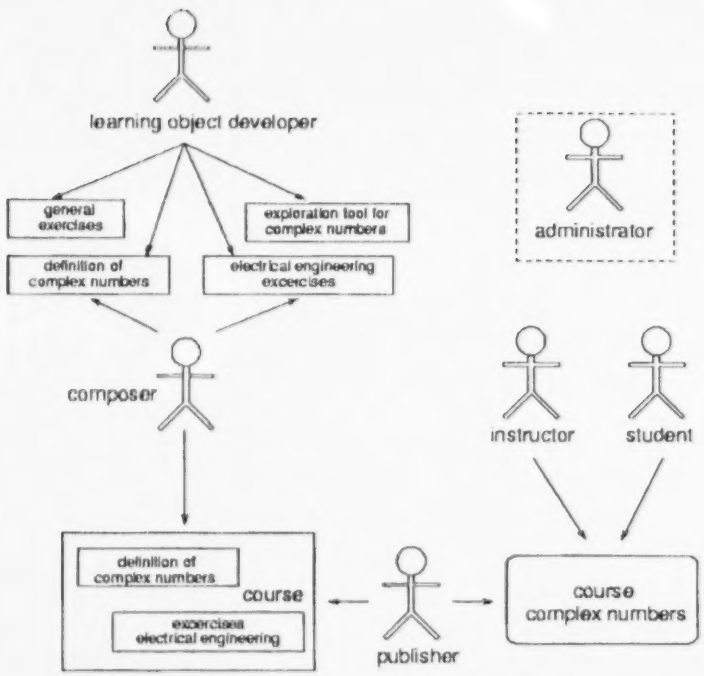

Figure 3: The roles that interact with the construction kit

After the course is completed, another person, such as a scientific assistant, takes on the role of the publisher. He can adjust the university emblem or any other style-related features like font, background color or page size. The product, a viewable document in a preferred format, is used by the professor, in the role of instructor, in his lecture to illustrate complex themes with animations. If students do not understand everything during the lecture they can read the complete course online at home. If a student needs more detailed information, for example about complex numbers, math-kit offers the whole database of content. Furthermore, the student is able to work through special exploration units which are provided by the professor as well.

The role of the administrator doesn't appear in this example because this role exists only to support the other roles.

Architecture

To understand the realisation of the construction kit it is necessary to have a look at its architecture. Usually, students use online courses through *Learning Management Systems (LMS)* such as LON-CAPA or Blackboard (LONCAPA 2002, BLACKBOARD 2002). A LMS provides different categories, for instance communicating to other students, sending solutions of exercises to lecturers, work through courses or use of multiple choice tests for check progress. The courses can be created with the help of authoring tools, for instance macromedia's authorware application. Unfortunately, these tools can only be used for universal content creation. They provide solutions for quizzes or give the author the possibility to adjust predefined design properties. In the math-kit project we need a special mechanism to define individual learning content. On the one hand, learning objects should be combined and transformed to individual courses. Thus, an authoring tool must provide the possibility to define learning objects without any style definition. On the other hand, different output formats should be provided as well. To meet this demand, the need of an abstract mechanism to define equations is required.

BEST COPY AVAILABLE

Based on these facts, we used an arbitrary LMS to provide the content of math-kit. This enables students and instructor to use their favored webbrowser to work through the courses or to select a special course block for presentation in lecture. However, the process of content creation must be supported by special tools in order to meet our requirements. According to the role concept, the learning object developer, the composer and the publisher are experts in content creation and each of them has his or her own tool. Related to our construction kit approach, these tools work with a server that stores courses and TCOs as bricks.

Thus, the *Constuction Kit Server* (CKS) is an important extension of existing LMS for individual content creation. Figure 4 shows the CKS and its tools.

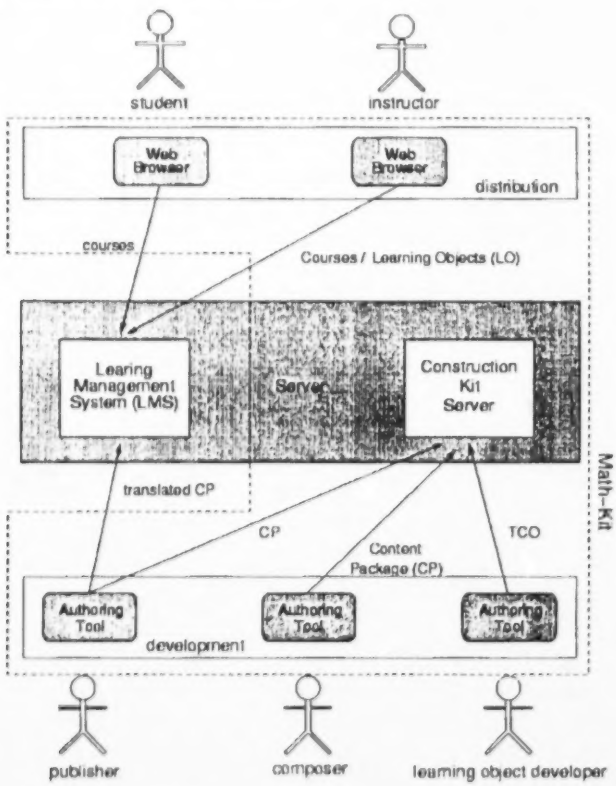

Figure 4: The architecture of the construction kit

The tools are split up into two groups: one for distribution and the other for development. The development tools are used to produce content packages that will be distributed to instructors and students through the LMS.

The tool for the *learning object developer* creates new TCOs that can be stored into the CKS. It enables the developer to create or modify templates easily by integrating new assets, for instance, images, texts or applets. As already mentioned, the TCOs usually content packages that consist of one XML-file with several referenced assets. The developer works on a copy of the original TCO file which is stored on the construction kit server. The client works on his or her own copy of the TCO. Thus, when a TCO is saved on the CKS the copy is merged with the original file. This is achieved by the use of CVS, a system to handle concurrent work.

For the *composer* role a separate authoring tool is available. This tool helps the process of combining TCOs to create the so called *content aggregation packages (CP)* (courses). Therefore a manifest file that contains the structure of the course has to be created. In relation to the construction-kit metaphor

BEST COPY AVAILABLE

the manifest is the construction plan. In Addition, the composer is able to search for existing TCOs by keywords based on the *case based reasoning (CBR)* (Watson 1997).

Another tool is available for the publisher for transforming content packages into courses that can be stored into different LMSs. With the help of this authoring tool, an individual design and layout can be defined. The CKS uses a stylesheet that is generated by the tool of the *publisher* which can transform a content package into a special output format, such as HTML or PDF. Using XSLT to transform source XML-files, the publisher has the option of choosing a predefined layout, modifying an existing layout or defining a new layout.

The consumer of the learning content doesn't pay attention to the CKS. They only have to access a LMS to work through courses or to use single learning objects for lectures.

Realisation

The implementation of the construction kit is influenced by the previously mentioned SCORM standard. All specifications in SCORM offer XML bindings so that a XML parser is needed. Because of platform independence, Java is our programming language of choice. Standard interfaces for XML processing in Java exist which are supported by many available parsers. For confidence we use JDOM (JDOM 2002), a simple XML document representation in Java with its own API that is similar to the W3C DOM (DOM 2002) but offers more comfort. With these tools it is quite simple to read, edit and write the manifest of a content package.

The authoring tool for the learning object developer allows the creation and modification of TCOs. This requires the handling of resource packages containing a template, assets and a manifest. To facilitate the work with TCOs, the direct manipulaton of assets should be provided. This gives the opportunity for a rapid implementation process controlled by one tool. The connection between the contained files and the tools must be realized via the *Multipurpose Internet Mail Extensions (MIME)* (MIME 2002). This specification defines among other things media types and subtypes to classify data. The application can determine the media type of a particular file in several ways, e.g. by the file extension or an analysis of the file beginning. Furthermore, every media type should be combined with an action, i.e. create, edit, or show.

A combination of media type and action is adhered to a Java component, called *JavaBean*, or an external application like a PDF viewer. Sun offers the *Java-Bean Activation Framework (JAF)* (Calder 1999) which supports the development of MIME media type driven applications. For better integration of self implemented or foreign JavaBeans a mechanism is required that allows dynamic and flexible adaption to personal needs. Java is a strong typed language and therefore programs have to be recompiled after every change. To get an acceptable solution, we recommend a scripting language as a bridge between the construction kit and the Java-Beans respectively external applications. An interpreter, called Jython, of the object-oriented scripting language, Python, exists for Java and is easy to integrate in applications. This combination of Java and Python provides all the benefits we want to offer with our appliation.

The composer role needs content packaging as well as search functionalities. Different TCOs have to be combined to higher structures, in detail content aggregation packages. Therefore, the right learning resources have to be offered by the authoring tool. For the realization of the case-based-reasoning searching, special data records have to be stored. The adequate solution to store these records is a relational database system. With the JDBC API (DOM 2002) of Java, a SQL driven implemetation could be done indepentendly of the concrete database system.

The publisher role needs support for the creation of styles but for most authors it is not reasonable to code XSLT files. Hence, the authoring tool offers an abstract view of the translation process. Every tag appearing in a template needs its own translation rule. To automate such a style process, a meta description for the templates is necessary. With this information, the authoring tool can support the publisher.

For the translation into readable output format, different rules and perhaps special pre- or post-processing components are required. Consider PDF as the desired output format. One solution could be a direct transformation into this format but this would result a huge programing effort. A more simple way to get PDF is the usage of *LaTeX* (Lamport, 1998). In this case, the template is translated into LaTeX code by XSLT and then into PDF by the post processor calling the LaTeX program.

Conclusion

We presented the architecture of an e-learning system that is based on the construction kit metaphor. With such a system, courses can be created by the usage of already existing learning resources based on our TCO approach. This approach makes it possible to adapt learning objects to different courses independent of their format and style. The architecture was developed in relation to the role concept. The definition of the roles working with the construction kit helps us to develop an architecture of an authoring system that can be used by students as well as professors to improve their work.

The universities of Hamburg, Hagen and Paderborn effectually use the construction kit to produce their courses (Padberg 2002) in the field of computer engineering as well as mechanical engineering.

References

ADL (2002). Sharable content object reference model (scorm). http://www.adlnet.org/.

Bergner K., Rausch A. & Sihling M. (1998) A critical look upon UML 1.0. In *The Unified Modeling Language– Technical Aspects and Applications*, pages 79–92.Physica-Verlag, Heidelberg.

Bungenstock M., Baudry A. & Mertsching B. (2002) The construction kit metaphor for a software engineering design of an e-learning system. In *ED-MEDIA 2002– World Conference on Educational Multimedia, Hypermedia & Telecommunications*, Denver, Colorado,USA.

BLACKBOARD (2002). B. Inc. The blackboard e-education. http://www.blackboard.com/.

Calder B. & Shannon B. (1999) Javabeans activation framework specification version 1.0a.

DOM (2002). Document object model (dom). http://www.w3.org/DOM/

IMS (2002). Ims content packaging specification. http://www.imsproject.org/content/packaging/.

JDOM (2002). http://www.jdom.org/

Lamport L. (1998). *LATEX A Document Preparation Syswtem User's Guide and Referencs Manual*. Addison-Wesley.

LONCAPA (2002). The learning online network with capa. http://www.lon-capa.org/

Mathkit (2002). http://www.math-kit.de.

MIME (2002). N. W. Group. Multipurpose internet mail extensions http://www.nacs.uci.edu/indiv/chood/MIME/2045/rfc2045.html.

Padberg K. & Schiller S. (2002). Web-based drills in maths using a computer algebra system. in Proc. EDMEDIA.

Unger L., Oevel G., Mertsching, B. (2002). Web-based teaching and learning with math-kit, 2th International Conference on the Teaching of Mathematics, Crete.

Watson I. D. (1997). *Applying Case-Based Reasoning : Techniques for Enterprise Systems*. M. Kaufmann.

XML (2002). W3C. Extensible markup language (xml) 1.0. http://www.w3.org/TR/2000/REC-xml-20001006.

XSLT (2002). W.W.W. Consortium. Xsl transformations (xslt) version 1.0. http://www.w3.org/TR/xslt.

U.S. Department of Education
Office of Educational Research and Improvement (OERI)
National Library of Education (NLE)
Educational Resources Information Center (ERIC)

NOTICE

Reproduction Basis

[X] This document is covered by a signed "Reproduction Release (Blanket)" form (on file within the ERIC system), encompassing all or classes of documents from its source organization and, therefore, does not require a "Specific Document" Release form.

[ ] This document is Federally-funded, or carries its own permission to reproduce, or is otherwise in the public domain and, therefore, may be reproduced by ERIC without a signed Reproduction Release form (either "Specific Document" or "Blanket").